Creating a test server

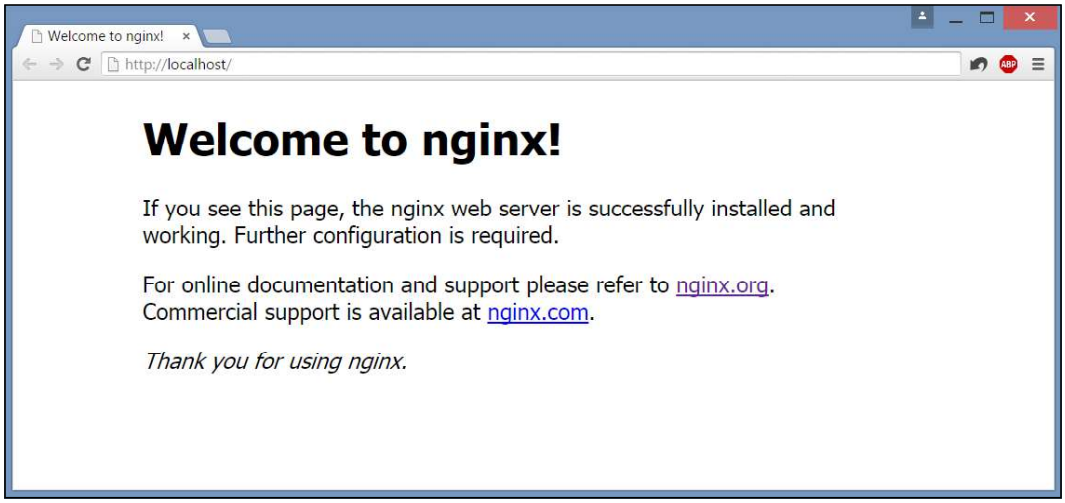
In order to perform simple tests, such as connecting to the server with a web browser, we need to set up a website for Nginx to serve. A test page comes with the default package in the `html` folder (`/usr/local/nginx/html/index.html`) and the original `nginx.conf` is configured to serve this page. The following is the section that we are interested in for now:

```
http {
    include       mime.types;
    default_type  application/octet-stream;
    sendfile        on;
    keepalive_timeout  65;
    server {
        listen       80;
        server_name  localhost;
        location / {
            root   html;
            index  index.html index.htm;
        }
        error_page   500 502 503 504  /50x.html;
        location = /50x.html {
            root   html;
        }
}
```

As you can perhaps already tell, this segment configures Nginx to serve a website:

- By opening a listening socket on port `80`
- Accessible at the address: `http://localhost/`
- With the index page, `index.html`

For more details about these directives, please refer to *Chapter 3, HTTP Configuration,* and go to the *HTTP module configuration* section. Anyhow, fire up your favorite web browser and visit `http://localhost/`:



You should be greeted with a welcome message; if you aren't, then check the configuration again, and make sure you reload Nginx in order to apply the changes.

Performance tests

Having configured the basic functioning and the architecture of your Nginx setup, you may want to proceed with running some tests. The methodology here is experimental—run the tests, edit the configuration, reload the server, run the tests again, edit the configuration again, and so on. Ideally, you should avoid running the testing tool on the same computer that is used to run Nginx, as it may give biased results.

One could question the pertinence of running performance tests at this stage. On one hand, virtual hosts and modules are not fully configured yet, and your website might use FastCGI applications (PHP, Python, and so on). On the other hand, we are testing the raw performance of the server without additional components (for example, to make sure that it fully makes use of all the CPU cores). Besides, it's always better to come up with a polished configuration before the server is put into production.